2016 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)

Timing Within Human-Agent Interaction and its Effects on Team Performance and Human Behavior

Tyler Goodman, Michael E. Miller, and Christina F. Rusnock
Systems Engineering and Management
Air Force Institute of Technology
Wright Patterson AFB, OH

Jason Bindewald
Department of Electrical and Computer Engineering
Air Force Institute of Technology
Wright Patterson AFB, OH

***Abstract*—Current systems incorporating human-agent interaction typically place the human in a supervisory role and the agent as a subordinate. However, a key aspect of teaming is the dynamic shift in roles. Depending on the situation at hand, teaming could lead to a peer relationship where the human and agent are working together on the same task. This research investigates how the timing of agent actions impacts team performance, as well as human workload and behavior. A human-in-the-loop experiment demonstrated that when the agent performs tasks faster than the human, the human tends to become reliant upon the automation and assumes a supervisory role. A human performance model predicts that extending agent execution time will decrease human reliance on the automation. However, in the environment under investigation, a tradeoff exists between team performance and human involvement.**

Keywords—Human-Machine Teaming; Agent; Trigger; Performance Modeling

I. Introduction

A. Human-Machine Teaming

The growing development and use of semi-autonomous systems has been beneficial in accomplishing tasks that would otherwise be error prone, dangerous, unmanageable, or simply impossible for humans [1]. Research efforts in this field have also increased in response to the rapid rise in technological capabilities. Significant and foundational pieces of literature have described autonomous systems as having several levels of automation when performing tasks typically allocated to a human operator [2,3]. This description of automation coincides with the design of several team-based descriptions of humans and autonomous coordination systems, including function allocation, supervisory control, adaptive automation, and dynamic task allocation [4].

One determining factor that separates a team from an ordinary group is a shared goal by all members [5] where cooperation is needed to limit interference between members during goal completion [6]. The purpose of teaming is to "increase the level of task performance by leveraging the unique capabilities of each performer, taking advantage of each member's strengths and available resources" [5]. Each team member's unique capabilities can help build individual alone [7]. To use each team member's strengths appropriately, teamwork is needed to facilitate interactions.

Current human-machine teams typically allocate responsibility such that the machine is subordinate to the human, thereby limiting the potential to which the team can leverage each member's unique strengths. Comparatively, effective human teams implement dynamic allocation of roles, responsibility, and authority dependent upon members' capabilities, availability, and task load. It is suggested that human-machine teams should model this schema to maximize performance in a dynamic environment. In classic systems, the machine usually fulfills the role of tool or subordinate, never reaching the status of a peer or leader. By allowing the machine to attain higher status, an emphasis on interdependence and communication emerges as each becomes more reliant upon the other [5].

Significant differences exist between humans and machines as team members. These differences not only include machine deficiencies, such as the limited ability to reason within the current context and respond to surprises in a robust manner [8], machine difficulty in communicating priorities [9], and lack of machine accountability [10]; but also seemingly pedestrian issues, such as ill-defined temporal requirements for operations.

The human information processing loop extending from perception through completion of an action often requires at least one third of a second and, depending upon the size of the muscle movements involved, can require multiple seconds. However, an agent, embedded in a computing system can perform a similar sequence of events in a much shorter period of time. Therefore, a designer may automate a process to improve system performance and decrease human workload. However, the incorporation of an automated tool can lead to the human adopting a supervisory role, which can be harmful to production. It has been documented that humans are poor monitors, a role they often assume when acting in a supervisory capacity, because they lose vigilance and are prone to fatigue [11]. Loss in vigilance can result in the human being "out of the loop", ultimately losing situation awareness. Consequently, it can be difficult for a person to understand the full context of a situation, possible actions, and consequences

The authors gratefully acknowledge the support of the Air Force Office of Scientific Research, Computational Cognition and Machine Intelligence Program, which partially funded this research.

if they have lost situation awareness when an unusual situation arises to which the automation cannot respond appropriately.

The idea of the human and machine working together as peers suggests that the human does not assume a supervisory role, but rather, the two are working alongside one another. There is a desire for the two to cooperate in such a manner where they are attaining adequate performance, yet the human is "in-the-loop" and maintaining situation awareness.

Triggers permit the automation to respond to events in the environment and actions by its team members. Triggers are developed to afford the automated system the ability to sense, observe, or model the environment to create a relative understanding of the events taking place around it and alter its behavior based upon this information. The goal of the automated agent is to receive relevant information from the environment and act accordingly [12]. Therefore, the trigger affects the automation's timing, i.e., time at which a task is initiated [10]. Logically, the timing of task execution in highly dynamic, event-driven domains must influence the performance and behavior of the team. Considering that automated systems have the potential to respond much faster than their human counterparts, their response time can affect task responsibility. If the automation's response time is too short, the human operator may assume the supervisory role as the automation will always respond to an event faster than its human counterpart. However, if its response is excessively delayed, the human is likely to assume responsibility for the event and attempt to respond before the automation. However, the proper timing and changes in the behavior of human team members as a function of automation response time is not apparent in the literature.

Therefore, this research aims to understand the effect of an automation's task timing on the performance of the human-machine team. This effect is examined using a combination of human-in-the-loop experimentation and human performance modeling within an environment employing an autonomous agent. A previous experiment is described which incorporated an autonomous agent that was triggered by the co-occurrence of an environmental event (i.e., appearance of a new task) and human inactivity in addressing this task [13]. The time frame at which the agent considered human inactivity to be excessive was static throughout the experiment. However, based upon the results of this experiment, it is assumed that variation in task timing of the automation will have a significant impact on user behavior. Thus, this research was conducted to explore the type of effects task timing has on team performance, as well as, human behavior and workload.

II. Method For Previous Experiment

A. Participants

The experiment involved 36 volunteers with an average age of 32.5 years and a range of 22 to 39 years. A total of 30 males and 6 females participated.

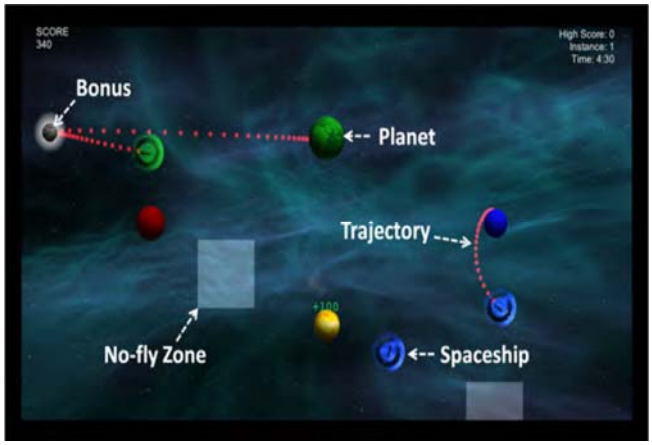
The experiment involved the use of a computer based tablet game environment. Thus, each participant was asked how often they use laptops, tablets, desktops, phones, and gaming consoles. On average, they used tablets roughly 1-3 times a week and gaming consoles 1-3 times a month. Other computer based platforms, including smart phones, were reported being used 3-7 times a week.

B. Apparatus and Environment

Space Navigator is a tablet-based computer trajectory-generation game which was constructed to provide a controlled representation of a highly-dynamic, event-driven environment. In these environments, the operator has little, if any, control of the event rate and there is no guarantee that the human will be capable of responding should unexpectedly high event rates occur. Similar environments might include air defense systems and certain command and control environments. The game, while not providing a high fidelity simulation of these environments, permits the control of the event rate and other potentially confounding variables, logging of human response, and the creation of automations that can be enabled to assist the operator during high event rate conditions. The use of the Space Navigator game for this study simplifies participant recruitment and training.

Figure 1 displays a screen capture from the game and identifies several key objects within the game. Spaceships appear at set intervals from the screen edges. The player directs each spaceship to its destination planet, designated through color, by drawing a line on the game screen using his or her finger. The spaceship then follows the entire drawn trajectory unless the player draws a different route for the ship. Points accumulate when a ship encounters its destination planet or one of a number of small bonuses that randomly appear throughout the play area. Points decrement when spaceships collide, and each spaceship involved in the collision is lost. Points are also lost when a spaceship traverses one of several "no-fly zones" that move to random locations within the play area at a set time interval. For every second a spaceship traverses a no-fly zone, the player loses points. The game ends after five minutes.

Fig. 1. Screen capture from Space Navigator, highlighting spaceships, planets, trajectories, bonuses, and no-fly zones.



In addition to drawing the routes manually, the subjects also work in human-agent teams in which both the subjects and the agents draw routes. There were three types of

automated agents: straight line, similar to the user, and dissimilar to the user. The straight line automation draws straight-line routes from the ship to the corresponding planet. The similar to the user automation uses a player model developed based on manual game play to draw routes predicted to be similar to those that the user would draw under similar circumstances. The dissimilar agent, selects random trajectories from the past game-play database. To provide the human with an opportunity to draw routes, the agent does not draw routes instantaneously, rather the automation triggers after a specified amount of on-screen time for a ship has elapsed without the subject interacting with that ship.

C. Experimental Design and Procedure

The experimental procedure consisted of a within subjects design in which each participant completed 16 five-minute instances of Space Navigator. The initial five instances contained no interaction from an automated agent and were used as participant training sessions. Following the training, participants completed three experimental sessions. Experimental sessions included four five-minute instances and each instance attributed one trajectory type to the agent throughout the entirety of a five-minute game. The four types of trajectories were either similar to the user, dissimilar to the user, straight line, and none (participant performed the task without an automated agent as a partner). Ships appeared on screen at a fixed rate of one ship appearing every two seconds. Bonuses and no-fly zones repopulated every thirty seconds.

D. Data Analysis

Game play and NASA-TLX [14] data were collected to assess user performance and workload per agent type. The Space Navigator environment actively stored information every time a ship-related action occurred. These actions included trajectory draws, collisions, bonus pickups, destinations reached, no-fly zone traversal, and off-screen movements. Subjective workload values were input by participants after completing each five-minute instance. Users were asked questions related to workload, frustration, and agent trust at the conclusion of the experiment. Although data was collected for three different agents, which performed differently from one another, the data analysis for the current paper was constrained to include only the manual condition in which there was no agent and the straight line agent, which drew a straight line from the ship to the appropriate planet anytime a ship resided on the screen for 2 s during which the participant did not draw a trajectory.

III. Experiment Results and Discussion

The expected result from this experiment was that the participants would continue drawing routes, relying on the automation to draw routes only when they were overloaded to the point that they could not draw routes quickly enough to be successful. The rationale behind this assumption was that this agent would be able to work alongside the user, but work less effectively and therefore not be trusted to draw routes unless the individual was task saturated to the point that they could not draw routes quickly enough. Therefore, it was expected that the majority of trajectories would be drawn by the participant. However, participants' behavior unanimously differed from this reasoning.

As shown in Table 1, when interacting with the game in a manual mode, without the agent, the human participants drew an average of 126.26 routes for the 150 ships that were generated during the 5 minutes of game play. Further, they redrew 21.83 routes for ships that they had already designated routes. However, when the straight line agent was employed, the humans drew less than $1/5^{th}$ as many trajectories on average (i.e., 23.19) than they did when playing the game manually. Additionally, when the agent was present, the participants redrew just over twice as many routes (mean of 43.97) as they did when operating in manual mode.

TABLE I. Manual and Straight Line Agent Data

	Fully Manual		Agent Assistance	
	Mean	St Dev	Mean	St Dev
Score	5801.57	2327.62	8043.06	1573.72
Hum. Draws	126.26	12.58	23.19	24.26
Redraws	21.83	11.94	43.97	15.55

Initially, it appeared counter-intuitive that the human participants would relinquish most of their initial path planning to an agent when the agent is incapable of making decisions based upon obvious obstructions or bonuses in the environment. However, this behavior becomes more understandable when one computes the average human ship-selection cycle-time. A full ship-selection cycle for the human involves identifying a ship to select, physically selecting a ship with their finger, and drawing a designated path. Analysis of this data reveals that an average of 2.6 s is required for a participant's ship-selection cycle-time whereas a new ship is spawned every 2 s. Therefore, it is implausible for the average human to successfully generate paths fast enough to provide a path for every ship. Conversely, the agent draws a route at the same speed as the ship spawn rate, drawing a route for the previously generated ship when the subsequent ship appears.

In this environment, with intuitive ship movement and a predictable agent, the participants were able to predict the behavior of the agent and then adjust undesirable paths. Consequently, it would appear that users began to initiate fewer trajectories, supervising the agent and redrawing paths to improve performance. As seen in Table 1, the addition of the agent increased the average score by roughly 2250 points (a 39% improvement) by having the human draw 103 fewer routes and doubling the number of redrawn routes.

Given this interaction, we therefore sought to better understand the interaction of the autonomous agent's timing within this environment. The trigger time employed in this experiment created an agent that assumed the human was overloaded if the human was unable to address an incoming ship within 2 s. It appeared that the automation's task timing exceeded the human operator's ability, relegating the operator to more of a supervisory role. Therefore, the participant game play data from this human-in-the-loop experiment was leveraged to construct a model of human-machine interaction.

The model was used to examine how variation in the automation's timing affects team performance, human behavior, and workload, within the teaming environment.

IV. Space Navigator IMPRINT Model

A. IMPRINT Simulation Software

To examine timing in the context of a human-machine team, this study uses the Improved Performance Research Integrated Tool (IMPRINT), a discrete-event simulation environment [15]. This environment models human workload and performance as a function of time by tracking activities performed by a human or a machine. These activities are described in a task network, which includes task sequencing and decision points. The frequency of the tasks, as well as the time necessary to perform each task result from a stochastic process, permitting the modeler to represent the variability within the system. Different task networks can be derived for different goals and a workload level is assigned to each task performed by the human operator. Various system allocations can then be modeled by allocating specific tasks to be performed by the human operator or machine (hardware or software). However, to employ this tool to accomplish this goal, the modeler must begin with activities to be performed by the team, allocate these activities to the human or machine and then derive the tasks or actions necessary to perform these activities. Once these activities are allocated to a component, human or machine, other inherent tasks may become necessary to facilitate communication of system state as control is passed between the human and machine[13, 16].

B. IMPRINT Task Network

The IMPRINT model is depicted in the SysML Activity Diagram [17] shown in Figure 2. This diagram divides the activities among three primary sections, separated by vertical lines known as "swim lanes", which separate the activities of the environment, the human operator, and the agent. The environment nodes in Space Navigator are responsible for starting the model, generating ships, altering no-fly zones and bonus locations, operating the timer, and halting the model as shown in the center "swim lane" of the activity diagram.

The player's attention and actions during game play are facilitated through a loop, continuously repeating two high level functions; determining which ship to select and drawing a trajectory for a ship. However, the loop is completed both for ships that have no drawn trajectories and for those that have a non-optimal trajectory. A view of ideal game play may include the person working to their capacity as they try to earn the highest score, leveraging the agent to draw paths they do not have time to draw. This behavior is depicted through the path in Figure 2 within the Human swim lane, which includes identifying background items, identifying ships without routes without waiting for the agent, selecting a ship and drawing a route. However, as demonstrated in the experiment, the human could permit the agent to draw some initial paths permitting them to attend to other tasks within the game. Thus, a task load node, indicated by the first decision node in the human swim lane, is used to represent a human's decision to either initiate ship selection or monitor the environment, allowing the human to observe the agent as it creates routes. The decision to monitor is based on a reliance algorithm derived from the experimental data, as seen in Figure 3.

The reliance algorithm produced a probability that the human would permit the agent to draw a trajectory. Analysis of the experimental data indicated that the probability of the agent drawing an initial route as a function of the number of ships on screen produced a parabolic curve. The participants performed more route draws when the number of ships on screen was low as they likely had ample time to interact with the system. They also appear to have drawn more routes when larger numbers of ships were on screen to help avoid collisions, given the agents' inability to react to neighboring ships, no-fly zones, and bonuses. The regression curve in Figure 3 accounted for the reliance of the operator on the automated agent with respect to the number of ships on screen.

The other factor that was necessary to include in the reliance algorithm was the trigger time of the agent. While no data exists to construct this function, it was assumed that the longer the agent takes before assigning a route, the more likely the human will initiate tasks to avoid losing points. At the lower limit, if the agent drew the line as soon as the ship appeared, the person would never have time to initiate a route. However, in the case that the agent requires an infinite amount of time before drawing the route, the human cannot rely upon the agent to draw any route. The operators' average cycle-time, time between initiating routes on separate ships, and standard deviation were derived from the experiment's fully manual gameplay. Using three standard deviations above and one standard deviation below the mean of 2.6 s, it was assumed that a human would be unlikely to initiate a route for a ship at 0.1 s and the agent would be unlikely to initiate a route at 11.6 s. This assumption was used to determine points on a linear equation relating delay time to probability of agent draws. This linear model was used to shift the third order regression line shown in Fig. 3 downwards as the agent's time delay increased and shift the regression line upwards as the agent's time delay decreased. For every second that the agent's delay changed, the baseline probability value was incremented or decremented by 0.1058, within the bounds that the probability must be between 0 and 1.

Returning to the task network, if the operator decides to draw a route based upon the reliance algorithm, they will continue to identify ships on screen. Afterwards, they can draw a route for a ship that does not have a route, or they can "redraw" a route for a ship that has an existing route. Following the draw route node, the human attention loops back to determine the background items, where the number of items impose a taskload, which is modeled as the number of ships on screen. As shown, when the human draws a route, the environment is updated, permitting both the agent to be informed by recording the route and displaying the route for the human.

Simultaneously, the agent is selecting ships and drawing trajectories for them as well, as indicated in the Agent swim lane of Figure 2. Unlike the human, the agent does not have

the option to perform fewer tasks. The agent is constantly monitoring all ships on screen and drawing a route once the time trigger has occurred. However, unlike the human, the agent can only draw trajectories for ships that have not received a trajectory, and the agent does not redraw non-optimal trajectories. As the agent draws a path, this information is provided to the environment.

Fig. 2. Activity diagram representing the actors and actions in the IMPRINT model. Vertical swimlanes are used to designate actions performed by the specified actor.

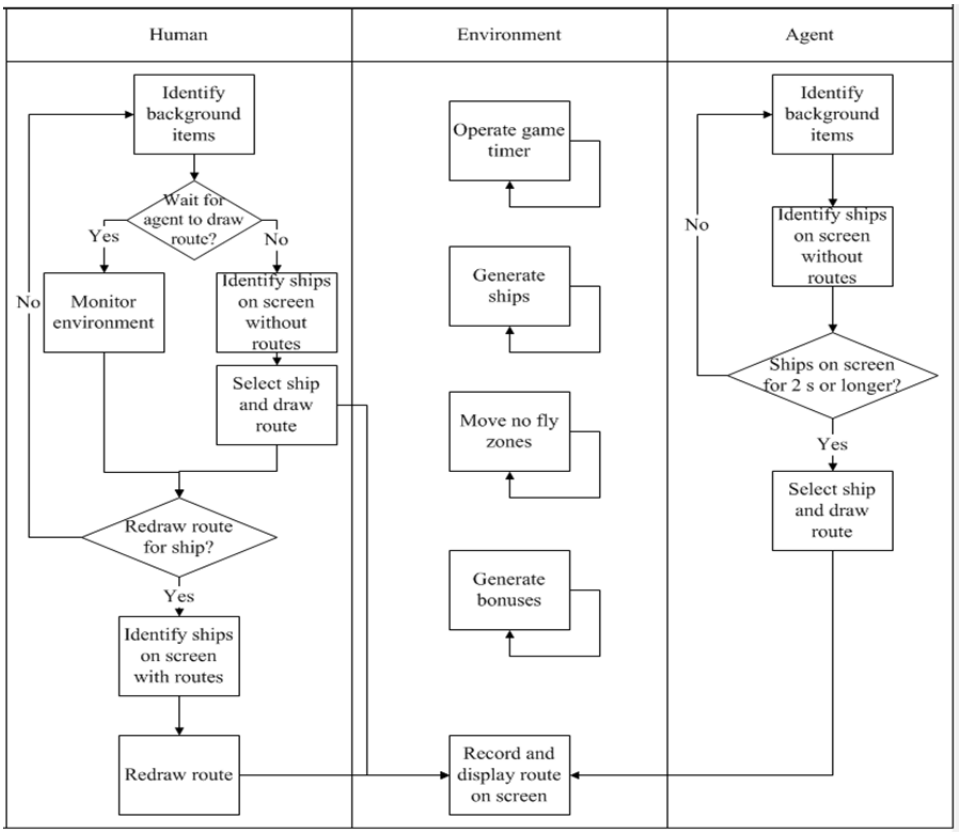
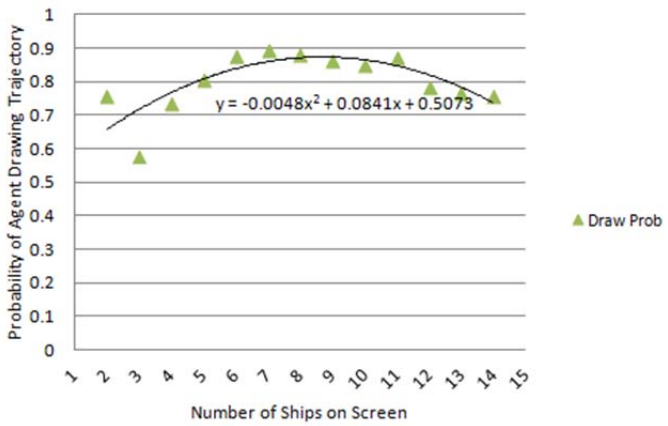


Fig. 3. Graph displaying the probability of the agent drawing a route with respect to the number of ships on the screen. The third order regression line, with equation, was used in calculating the reliance algorithm in the IMPRINT model.



After the human or agent has designated a route for a ship, a new entity is created in the model, representing the ship with a route. The ship continues along its path for a length of time drawn from a distribution representing game play time-on-screen and is be removed from the simulation after the time has elapsed (not depicted in Figure 2). There are three possible end results for a ship: collision, destination reached, and off-screen traversal. Ships arrive to these nodes according to probabilities associated with the number of ships on screen and the human or agent that drew the route. Once again these distributions are developed from the human-in-the-loop data discussed earlier.

C. Model Validation

To validate the model, the model was exercised for conditions that matched the conditions of the previously explained human-subjects experiment and the results were compared. The IMPRINT model replicated the experimental trials by having the agent create routes for ships that were on screen, and without a route, for two seconds or longer. The results applied for model validation were scores, number of automation trajectories drawn, and number of "redrawn" trajectories by the operator. These specific aspects of the model were chosen to ensure that performance and behavior,

as predicted by the model, was similar to the data from the human-in-the-loop experiment. To compare score values and trajectories, two sample t-tests with 95% confidence intervals were performed. For score, the average from the experiment was 8043 (sd 1574) while the mean from the model was 8053 (sd 871). The t-test indicated that these values were not statistically different ($t(1,169 = -0.06$, $p=0.955$). The average number of agent-drawn trajectories from the experiment was 126.9 (sd 24.4) and the mean from the model was 122.4 (sd 3.24). The t-test indicated that these values were not statistically different ($t(1,109) = -1.91$, $p=0.06$). The average number of human redraws from the experiment was 44 (sd 15.5) and the mean from the model was 45.59 (sd 6.08). The t-test indicated that these values were not statistically different ($t(1,141) = -1.00$, $p=0.318$). Overall, there was no evidence of statistical differences between the model and the experimental data, and thus the model is considered validated.

The workload values collected in the human-subjects experiment were NASA-TLX values, whereas the workload inputs in IMPRINT are from the Visual, Auditory, Cognitive, and Psychomotor (VACP) workload assessment tool. Consequently, workload could not be directly validated. Thus, validation was conducted with a subject matter expert.

As the slope of the linear equation relating agent delay time and probability of an agent draw was assumed during model construction, it is important to understand the sensitivity of the model to this slope. Simulations were run with a 10% increase and decrease to this slope. At the lower bound, on average the human drew 2.5 fewer trajectories and scored 24 fewer points. At the upper bound, the average score increased by 18 points and the human drew 2.14 more trajectories. The change in both values was greatest during the 8.6 s delay time where in the lower bound the human drew 10 fewer trajectories and in the upper bound the human drew 8.35 more trajectories. The difference in score for both fluctuated and had no distinct pattern. The change in workload and redraws was negligible. Therefore, it is believed that changes in this slope will significantly affect the model results for delay times near the intersection of the linear model with the delay time axis. However, the characteristic shape of model output as a function of timing delay is likely to be robust.

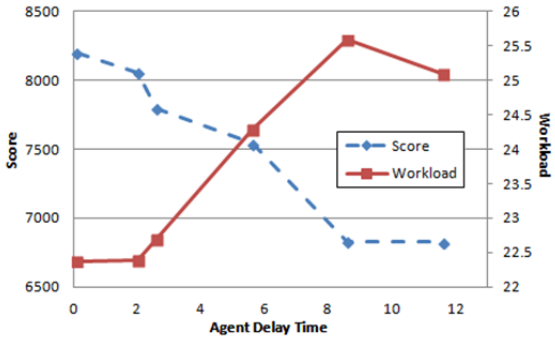
D. Simulation Procedure

A series of simulations were conducted in which the trigger time of the automated agent was altered in each simulation. Trigger times were selected based upon participant performance. As noted earlier, the participant required an average of 2.6 seconds between the time a ship is spawned, appearing on screen, and the time the human selects the ship to draw a trajectory. The associated standard deviation of this time was 3.0 seconds. Six conditions are evaluated: the mean time for a participant to select a ship (2.6 s), plus one-half, one, two and three standard deviations (ie., 0.1, 5.6, 8.6 and 11.6 s), as well as the original 2 s delay employed in the human-in-the-loop experiment. The six scenarios were each simulated 100 times, having the same random seed value for each condition. At the end of each scenario, the average scores, workload, and trajectories drawn were calculated. A one-way Analysis of Variance (ANOVA) was used to determine whether agent delay-time had a significant effect on any of the model outputs and Tukey Pairwise Comparisons were used to test for differences between individual means.

V. Simulation Results

The results from the IMPRINT simulations displayed an inverse relationship between performance and workload, as shown in Figure 4. As the trigger time increased beyond the average time of 2.6 s, the operator's workload increased and overall team performance decreased. Although performance, in terms of overall score, was recorded for each of 100 model runs, workload is shown for a typical single model run.

Fig. 4. Graph displaying average score and workload per agent delay time

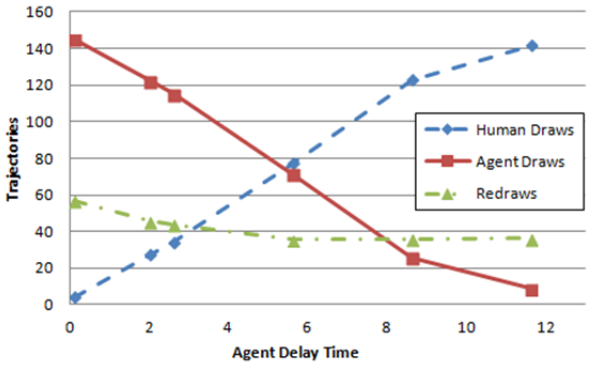


The ANOVA indicated that the effect of agent trigger time on overall score is statistically significant ($F(5,594) = 43.28$; $p < 0.001$). Tukey pair-wise comparisons indicated that there were four groups of scores that were significantly different from one another. These groups in terms of agent time delay were (0.1, 2.0), (2.0, 2.6), (2.6, 5.6) and (8.6, 11.6). It was shown in these pairings that as the time delay increased the average score significantly decreased.

As shown in Figure 5, the human and agent draws were also inversely related. The agent's trigger time significantly affected agent draws ($F(5,594) = 35784$; $p < 0.001$), human draws ($F(5,594) = 31975$; $p < 0.001$), and redraws ($F(5,594) = 174$; $p < 0.001$). The ANOVA for agent and human draws produced similar results. The number of human draws were statistically different for all agent redraw conditions. The effect of time on redraws generated three different groupings, with 0.1 s producing the most redraws, followed by 2.0 and 2.6 s conditions and 5.6, 8.6, 11.6 s conditions.

These simulations indicate that human behavior will change as a function of the agent's trigger time. When the agent created routes at the same speed or faster than the human, the human initiated routes between 2% and 20% of the time. When the trigger time is one to three standard deviations slower, the number of human initiated routes increased from 50% to 95%. Furthermore, the model anticipated that the largest shift in performance would occur when the trigger time was adjusted from 5.6 to 8.6 s, decreasing the score by 10%. The greatest increases in workload should occur when delay times change from 2.6 to 5.6 s and 5.6 to 8.6 s, with a 7% and 5% increase, respectively.

Fig. 5. Graph displaying mean human draws, agent draws, and redraws per agent delay time



VI. Conclusions and Future Work

According to the simulations, timing of the interaction between the human and automated agent significantly affects system performance, human workload, and behavior. As a result, the agent's task time can be determined to support system objectives. For example, if the only objective is to obtain the highest score possible, it seems appropriate to place the agent trigger at 0.1 s to obtain the best possible team score. However, if there is an added objective, such as keeping the user engaged in drawing a portion of the initial routes, the approach should vary. For the operator to respond correctly to any error that might occur, they need to detect and understand the context of the error. By having the agent trigger too quickly, the human is likely to learn to redraw paths without drawing initial paths. Under conditions of low task load, as might occur as the spawn rate of the ships is reduced, the user may fall into performing a vigilance task and potentially lose the ability to maintain situation awareness. Therefore, while it may be optimal to have a quick trigger to earn higher points, this same trigger could be detrimental if the human is unable to maintain alertness and therefore be unable to detect agent errors. The purpose of keeping an operator "in the loop" is to ensure they are capable of making appropriate decisions when tasked accordingly. Keeping the operator "in the loop" appears to correlate with the timing of human-agent interaction.

Future studies could investigate how an individual's tendency to trust an automated agent affects performance, workload, and behavior. This can be evaluated by adjusting the reliance function to represent varying levels of trust. Furthermore, the results from this model suggest that human-subjects experiments should be performed to validate the behavior predicted in this research. If those experiments affirm this research, it could provide insight into the significance of timing when human and agents work together on the same task. Finally, one might expect that agent delay time is not only dependent upon the human's response time but also upon the taskload modulated as a function of ship spawn rate. Understanding interactions between these variables may provide an understanding of human-machine teaming based upon agent timing.

Acknowledgment

The authors would like to thank Jayson Boubin for his contributions to model development and Dr. Gilbert Peterson for assisting in direction of agent development.

The views in this article are those of the authors and do not necessarily reflect the official policy or position of the Department of the Air Force, Department of Defense, nor the U.S. Government.

References

[1] P. Millot, *Designing Human-Machine Cooperation Systems*. London, UK: John Wiley & Sons, 2014.

[2] R. Parasuraman, T. B. Sheridan, and C. D. Wickens, "A model for types and levels of human interaction with automation.," *IEEE Trans. Syst. Man. Cybern. A Syst. Hum.*, 30(3), pp. 286–297, 2000.

[3] D. B. Endsley, M.R., and Kaber, "Level of Automation Effects on Performance, Situation Awareness and Workload in a Dynamic Control Task," *Ergonomics*, 42, pp. 462–492, 1999.

[4] M. Johnson, J. M. Bradshaw, P. J. Feltovich, R. R. Hoffman, C. Jonker, B. Van Riemsdijk, and M. Sierhuis, "Beyond cooperative robotics: The central role of interdependence in coactive design," *IEEE Intell. Syst.*, 26(3), pp. 81–88, 2011.

[5] D. J. Bruemmer, J. L. Marble, and D. D. Dudenhoeffer, "Mutual initiative in human-machine teams," in *Proceedings of the IEEE 7th Conference on Human Factors and Power Plants*, 2002, pp. 22–30.

[6] J.-M. Hoc, "Towards a cognitive approach to human–machine cooperation in dynamic situations," *Int. J. Hum. Comput. Stud.*, 54(4), pp. 509–540, 2001.

[7] W. Arthur, B. D. Edwards, S. T. Bell, A. J. Villado, C. Station, and W. Bennett, "Team Task Analysis : Identifying Tasks and Jobs That Are Team Based," 47(3), pp. 654–669, 2005.

[8] T. Huntsberger, "Cognitive architecture for mixed human-machine team interactions for space exploration," *IEEE Aerosp. Conf. Proc.*, 2011.

[9] G. Klein, D. D. Woods, J. M. Bradshaw, R. R. Hoffman, and P. J. Feltovich, "Ten challenges for making automation a 'team player' in joint human-agent activity," *IEEE Intell. Syst.*, 19(6), pp. 91–95, 2004.

[10] M. Anderson, S. L. Anderson, and C. Armen, "Towards Machine Ethics," in *Proceedings of the AAAI 2005 Fall Symposium on Machine Ethics*, 2004, pp. 1–7.

[11] R. Parasuraman, "Supporting Battle Management Command and Control: Designing Innovative Interfaces and Selecting Skilled Operators," Fairfax, VA, 2008.

[12] K. M. Feigh, M. C. Dorneich, and C. C. Hayes, "Toward a Characterization of Adaptive Systems: A Framework for Researchers and System Designers," *Hum. Factors*, 2012.

[13] J. M. Bindewald, M. E. Miller, and G. L. Peterson, "A function-to-task process model for adaptive automation system design," *Int. J. Hum. Comput. Stud.*, vol. 72, no. 12, pp. 822–834, 2014.

[14] S. G. Hart and L. E. Staveland, "Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research," *Adv. Psychol.*, 52(C), pp. 139–183, 1988.

[15] "Improved Performance Research Integration (IMPRINT) Tool," *Army Research Laboratory*, 2010. [Online]. Available: https://dap.dau.mil/aphome/das/Lists/Software Tools/DispForm.aspx?ID=58.

[16] T. Goodman, M. Miller, and C. Rusnock, "Incorporating Automation: Using Modeling and Simulation to Enable Task Re-Allocation," in *Winter Simulation Conference*, 2015.

[17] L. Delligatti, *SysML Distilled: A Brief Guide to the Systems Modeling Language*. Addison-Wesley, 2013.